Amendment to the Specification:

Rewrite the first paragraph on page 1 to read as follows:

This application claims priority under 35 USC §119(e)(1) of Provisional Application No. 60/183,527, filed February 18, 2000 (TI-30302PS) and of Provisional Application No. 60/173,761, filed December 30, 1999 (TI-26011P); and is related to copending application S.N. 09/703,096 filed October 31, 2000, which claims priority of Provisional Application No. 60/183,527; and is related to copending application S.N. 09/687,540 filed October 13, 2000, which claims priority of Provisional Application No. 60/173,761.

Rewrite the second full paragraph on page 8 to read as follows:

In microprocessor 1 there are shown a central processing unit (CPU) 10, data memory 22, program memory 23, peripherals 60, and an external memory interface (EMIF) with a direct memory access (DMA) 61, coupled to data memory 22 via lines 43. CPU 10 further has an instruction fetch/decode unit 10a-c, a plurality of execution units, including an arithmetic and load/store unit D1, a multiplier M1, an ALU/shifter unit S1, an arithmetic logic unit ("ALU") L1, a shared multiport register file 20a from which data are read and to which data are written. Instructions are fetched by fetch unit 10a from instruction memory 23 over a set of busses 41. Decoded instructions are provided from the instruction fetch/decode unit 10a-c to the functional units D1, M1, S1, and L1 over various sets of control lines which are not shown. Data are provided to/from the register file 20a from/to to load/store units D1 over a first set of busses 32a, to multiplier M1 over a second set of busses 34a, to ALU/shifter unit S1 over a third set of busses 36a and to ALU L1 over a fourth set of busses 38a. Data are provided to/from the memory 22 from/to the load/store units D1 via a fifth set of busses 40a. Note that the entire data path described above is duplicated with register file 20b and execution units D2, M2, S2, and L2. In this embodiment of the present invention, two unrelated aligned double word (64 bits) load/store transfers can be made in parallel between CPU 10 and data memory 22 on each clock cycle using bus set 40a and bus set 40b.

Rewrite the third full paragraph on page 10 to read as follows:

A detailed description of various architectural features of the microprocessor of Figure 1 is provided in coassigned application S.N. 09/012,813 ~~(TI-25311)~~, now U.S. Patent No. 6,182,203, and is incorporated herein by reference. A description of enhanced architectural features and an extended instruction set not described herein for CPU 10 is provided in coassigned U.S. Patent application S.N. 09/703,096 ~~(TI-30302)~~ *Microprocessor with Improved Instruction Set Architecture* and is incorporated herein by reference.

Rewrite the first full paragraph on page 14 to read as follows:

Each functional unit reads directly from and writes directly to the register file within its own data path. That is, ~~the~~ .L1 unit 18a, .S1 unit 16a, .D1 unit 12a, and .M1 unit 14a ~~units~~ write to register file A and ~~the~~ .L2 unit 18b, .S2 unit 16b, .D2 unit 12b, and .M2 unit 14b ~~units~~ write to register file B. The register files are connected to the opposite-side register file's functional units via the 1X and 2X cross paths. These cross paths allow functional units from one data path to access a 32-bit operand from the opposite side's register file. The 1X cross path 210 allows data path A's functional units to read their source from register file B, via multiplexers 211, 212, 213, 214. Similarly, the 2X cross path allows data path B's functional units to read their source from register file A.

Rewrite the second full paragraph on page 14 to read as follows:

All eight of the functional units have access to the opposite side's register file via a cross path. The .M1, .M2, .S1, .S2, .D1 and .D2 units' *src2* inputs are selectable between the cross path and the same side register file. In the case of the .L1 and .L2 units 18a, 18b, both *src1* and *src2* inputs are also selectable between the cross path and the same-side register file, for example by operation of multiplexers 211, 212 as shown in Figure 2 in connection with .L1 unit 18a. In addition, as shown in Table 2, transfers to control register file 102 over bus 220 and from control register file 102 over bus 221, may also be made via .S2 unit 16b only.

Rewrite the first full paragraph on page 15 to read as follows:

A delay clock cycle is introduced whenever an instruction attempts to read a register via a cross path that was updated in the previous cycle. This is known as a cross path stall. This stall is inserted automatically by the hardware, as suggested by delay block 250 in Figure 2; no NOP instruction is needed. It should be noted that no stall is introduced if the register being read is the destination for data loaded by a LDx instruction.

Rewrite the third full paragraph on page 16 to read as follows:

Table 3 defines the mapping between instructions and functional units for a set of basic instructions included in a DSP described in U.S. Patent Application S.N. 09/012,813, now U.S. Patent No. 6,182,203 (~~TI-25311,~~ incorporated herein by reference). Table 4 defines a mapping between instructions and functional units for a set of extended instructions in an embodiment of the present invention. Alternative embodiments of the present invention may have different sets of instructions and functional unit mapping. Table 3 and Table 4 are illustrative and are not exhaustive or intended to limit various embodiments of the present invention.

Rewrite the first full paragraph on page 19 to read as follows:

The addressing modes for instructions that access memory ~~are~~ may be linear, or circular ~~using BK0, and circular using BK1~~. The mode is specified by an addressing mode register (AMR) contained in control register file 102. Eight registers can perform circular addressing. A4-A7 are used by the .D1 unit and B4-B7 are used by the .D2 unit. No other units can perform circular addressing modes. For each of these registers, the AMR specifies the addressing mode.

Rewrite the third full paragraph on page 21 to read as follows:

Performance can be inhibited by stalls from the memory system, stalls for cross path dependencies, or interrupts. The reasons for memory stalls are determined by the memory architecture. Cross path stalls are described in detail in U.S. Patent S.N. 09/702,453, filed October 31, 2000 (~~TI-30563~~), to Steiss, et al and is incorporated herein by reference. To fully understand how to optimize a program for speed, the sequence of program fetch, data store,

and data load requests the program makes, and how they might stall the CPU should be understood.

Rewrite the first full paragraph on page 27 to read as follows:

Figure 5A illustrates an instruction syntax for a byte intermingling instructions that selects byte fields from both a first source operand and from a second source operand, such as the Shift Left, Merge Byte (SHLMB) instruction. Figure 5B illustrates an instruction syntax for a byte intermingling instructions that selects byte fields from only one source operand, as the SWAP4 instruction. In this embodiment, all of the byte intermingling instructions can be executed in either .L functional unit 18a or 18b as indicated by unit select bit field 500. The instruction includes a first source operand field (src1) 501 and a second source operand field (src2) 502 that each select a register from associated register file 20a or 20b to access a source operand which is a 32-bit data value. The byte intermingling instructions each perform a byte intermingling operation on various fields selected from the source operands. The values in the source operands are treated as packed data, and the result is written in a corresponding packed format in a destination register specified by a destination field (dst) 504. Each of the byte intermingling instructions in this embodiment, except SWAP4, PACKH4 and PACKL4, can also be executed on either .S unit in response to a different value in type field 510 and opcode field 512.

Rewrite the carryover paragraph from page 28 to page 29 to read as follows:

Figure 6A is a flow chart illustrating operation of a Shift Left and Merge Byte (SHLMB) instruction. The SHLMB instruction shifts the contents of *src2* left by one byte, and then the most significant byte of *src1* is merged into the least significant byte position. The result is then placed in *dst*. A data value in a first selected source operand 600 is treated as packed, unsigned 8-bit data, located in four distinct fields 600(0-3). A data value in a second selected source operand 602 is also treated as packed unsigned 8-bit data, located in four distinct fields 602(0-3). Three fields 602(2-0) are selected from a least significant portion of the second source operand 602 and shifted left and placed in order in fields 610(3-1) of destination operand 610. A most

significant field 600(3) is selected from the first source operand 600 and placed in a least significant position 610(0) in destination operand 610. Thus, a destination operand is formed that has unsigned bytes selected from the first operand (ua_n) and from the second operand (ub_n) in the following order: ub_2, ub_1, ub_0, and ua_3. In this embodiment, the destination is written during pipeline phase E1 and the SHLMB instruction is categorized has having no delay slots.

Rewrite the third full paragraph on page 31 to read as follows:

Figure 7A is a top level block diagram of .L unit 18a or 18b, which is optimized to handle logical operations, although hardware is available for a set of add and subtract operations and also for the multi-field intermingling instruction of the present invention. Logic block 700 performs various Boolean logic functions. Pass gates 700a together with keeper gates 700b form a latch to hold the contents of a first source operand src1, which is selected from either register file 20a or 20b via mux 211 (see Figure 2). Similarly, pass gates 700c ~~700ea~~ together with keeper gates 700d form a latch to hold the contents of a second source operand src2, which is selected from either register file 20a or 20b via mux 212 (see Figure 2).

Rewrite the carryover paragraph from page 32 to page 33 to read as follows:

Figure 7B is a more detailed block diagram of intermingling circuit 702 of Figure 7A. Four separately controlled 8-bit multiplexers 730(3:0) are each connected to receive all thirty two bits of source operands src1 and src2, along with logical zeros and logical ones. Thus, any combination of byte fields can be selected from the two source operands and intermingled in any order to form a destination operand on output signal lines mux(31:0). Mux control circuitry 732 receives signals 734 from instruction decoding circuitry 10c (see Figure 1) that indicate which byte intermingling instruction is being executed. Separate sets of control signals 732(3:0) are sent to each multiplexer to select to appropriate byte fields from src1 and src2 in response to the byte intermingling instruction that is being executed in order to form a destination operand.

Rewrite the first full paragraph on page 33 to read as follows:

Figure 7C is an alternate embodiment of an intermingling circuit. Four separately controlled 8-bit multiplexers 740(3:0) are each connected to receive only the byte fields of source operands src1 and src2, along with logical zeros, needed to form the intermingled destination operands as indicated in Table 11. Thus, sets of byte fields can be selected from the two source operands and intermingled to form a destination operand on output signal lines mux(31:0) according to the set of byte intermingling instructions described in Table 9. Mux control circuitry 742 receives signals 744 from instruction decoding circuitry 10c (see Figure 1) that indicate which byte intermingling instruction is being executed. Separate sets of control signals 742(3:0) are sent to each multiplexer to select to appropriate byte fields from src1 and src2 in response to the byte intermingling instruction that is being executed in order to form a destination operand.

Rewrite the first full paragraph on page 34 to read as follows:

In another embodiment of the invention, an intermingled destination operand may be written directly to memory rather to a register file. This is described in more detail in Co-assigned U.S. Patent application S.N. 09/687,540 ~~(TI-26011)~~ entitled *Data Processing System with Register Store/Load Utilizing Data Packing/Unpacking* and is incorporated herein by reference.

Rewrite the second full paragraph on page 34 to read as follows:

Figure 8 is a block diagram of an alternative embodiment of the present invention in a digital system 1000 including integrated circuit 1001 with a processor core ~~10-of~~ corresponding to that described above relative to Figure 1. A direct mapped program cache 1010b, having 16 kbytes capacity, is controlled by L1 Program (L1P) controller 1010a ~~1011~~ and connected thereby to the instruction fetch stage 10a. A 2-way set associative data cache 1020b, having a 16 Kbyte capacity, is controlled by L1 Data (L1D) controller 1020a ~~1721~~ and connected thereby to data units D1 and D2. An L2 memory 1030 having four banks of memory, 128 Kbytes total, is connected to L1P 1010a ~~1011~~ and to L1D 1020a ~~1021~~ to provide storage for data and programs. External memory interface (EMIF) 1050 provides a 64 bit data path to external memory, not

shown, which provides memory data to L2 memory 1030 via extended direct memory access (DMA) controller 1040.

Rewrite the carryover paragraph from page 34 to page 35 to read as follows:

Three multi-channel buffered serial ports (McBSP) 1060, 1062, 1064 are connected to DMA controller 1040. A detailed description of a McBSP is provided in U.S. Patent S.N. 09/055,011, now U.S. Patent No. 6,167,466 ~~(TI-26204, Seshan, et al)~~ and is incorporated herein by reference.

Rewrite the Abstract of the Disclosure to read as follows:

A data processing system is provided with a digital signal processor that has a set of instructions for intermingling byte fields selected from a selected pair of source operands and storing the ordered result in a selected destination register. A first 32-bit operand ~~(600)~~ is treated as four 8-bit fields while a second 32-bit operand ~~(602)~~ is treated as four 8-bit fields. Intermingling circuitry ~~702~~ is operable to form an ordered result in accordance with each one of the set of byte intermingling instructions. An instruction is provided that performs a shift right and byte merge operation. Another instruction is provided that performs a shift left and byte merge operation. Another instruction is provided that perform a byte swap operation. A set of instructions are provided that perform various byte packing and unpacking operations.

~~Figure 6A-6L~~